

No need to choose: How to get both a PTAS and Sublinear Query Complexity

Nir Ailon¹ and Zohar Karnin²

¹ Technion IIT and Yahoo! Research, Haifa, Israel nailon@cs.technion.ac.il

² Yahoo! Research, Haifa, Israel zkarnin@yahoo-inc.com

Abstract. We revisit various PTAS's (Polynomial Time Approximation Schemes) for minimization versions of dense problems, and show that they can be performed with sublinear query complexity. This means that not only do we obtain a $(1 + \varepsilon)$ -approximation to the NP-Hard problems in polynomial time, but also avoid reading the entire input. This setting is particularly advantageous when the price of reading parts of the input is high, as is the case, for examples, where humans provide the input. Trading off query complexity with approximation is the *raison d'être* of the field of learning theory, and of the ERM (Empirical Risk Minimization) setting in particular. A typical ERM result, however, does not deal with computational complexity. We discuss two particular problems for which (a) it has already been shown that sublinear querying is sufficient for obtaining a $(1 + \varepsilon)$ -approximation using unlimited computational power (an ERM result), and (b) with full access to input, we could get a $(1 + \varepsilon)$ -approximation in polynomial time (a PTAS). Here we show that neither benefit need be sacrificed. We get a PTAS with efficient query complexity.

The first problem is known as Minimal Feedback Arc-Set in Tournaments (MFAST). A PTAS has been discovered by Schudy and Mathieu, and an ERM result by Ailon. The second is k -Correlation Clustering (k -CC). A PTAS has been discovered by Giotis and Guruswami, and an ERM result by Ailon and Begleiter.

Two techniques are developed. The first solves the problem for the low-cost case of k -CC (the analogous case is already known for MFAST). This requires a careful sampling scheme together with proof of a structural property relating costs of vertices against the optimal sample clustering with their costs against the full optimal clustering. The second addresses the high-cost case, by showing that a classic method by Arora et al. (2002) for obtaining additive approximations can be made query efficient. The underlying technique is “double sampling”: One sample is amenable to exhaustive solution enumeration, but well approximates only polynomially many solutions (including the optimal), and another sample cannot be used exhaustively search solutions, but well approximates the cost of the entire solution space, and is used for verification.

1 Introduction

We study two NP-Hard combinatorial minimization problems for which it is known how to get a $(1 + \varepsilon)$ -approximate solution under two scenarios. In the

first scenario, the algorithm has full access to the input, and is required to compute in polynomial time. In the second scenario, the algorithm has exponential computational power but is allowed to uncover only a sublinear amount of input. In this work we show that no requirement needs to be sacrificed. In other words, we satisfy the following three requirements simultaneously:

- (R1) A polynomial time algorithm.
- (R2) A $(1 + \varepsilon)$ approximate solution.
- (R3) A sublinear (in input size) query complexity.

The first problem is known as k -Correlation Clustering (k -CC). Given an undirected graph $G = (V, E)$, the objective is to find a decomposition of V into k (possibly empty) disjoint subsets (clusters) C_1, \dots, C_k so that the symmetric difference between E and the set $\{(u, v) : \exists i \text{ s.t. } \{u, v\} \subseteq C_i\}$ is minimized. The second problem is the Minimum Feedback Arc-set in Tournaments (MFAST). In this problem, given a tournament $G = (V, A)$, the objective is to write its vertices in a sequence from left to right so that the number of edges pointing to the left (*backward edges*) is minimized.³ Requirements (R1) and (R2) are achieved by Giotis et. al in [9] for k -CC and by Kenyon-Mathieu et. al in [10] for MFAST. Requirements (R2) and (R3) were achieved very recently by Ailon et. al in [2] for k -CC and by Ailon in [1] of MFAST. In this work we obtain (R1)+(R2)+(R3) for both problems. Our result uses components from the aforementioned citations, together with new ideas required for obtaining our strong guarantees.

1.1 Previous Work and Our Contribution

In the world of combinatorial approximations, Correlation Clustering (CC) (also known as cluster editing) has been defined by Blum et al. [6]. In the original version there was no bound on the number k of clusters. Correlation clustering is max-SNP-Hard [8] but admits constant factor polynomial times approximations (e.g. [8,3]). Maximization versions have also been considered [11]. In this work we concentrate on the minimization problem only, which is more difficult for the purpose of obtaining a PTAS. The k -correlation clustering (k -CC), in which the number of output clusters is bounded by k , is also NP-Hard but admits a PTAS [9] running in time $n^{O(9^k/\varepsilon^2)} \log n$. There is a natural machine learning theoretical interpretation CC: The instance space is identified with the space of element pairs, and each edge (resp. non-edge) in G is a label stipulating equivalence (resp. non-equivalence) of the corresponding pair. The CC cost minimizes the *risk*, defined as the number of pairs of elements on which the solution disagrees with. Roughly speaking, an algorithm attempting to minimize the risk by, instead, minimizing an estimator thereof obtained by sampling labels, is an *Empirical Risk Minimization* (ERM) algorithm. An ERM algorithm need not be constrained by computational restrictions, and should be thought of as a *information theoretical*, not computational result. It should be noted that machine learning clustering theoreticians and practitioners have been studying how

³ By tournament we mean that for all distinct $u, v \in V$, either $(u, v) \in A$ or $(v, u) \in A$.

to use correlation clustering type labels in conjunctions with more traditional geometric clustering approaches (e.g. k -means - see Basu's thesis [7] and references therein). Such labels are expensive because they require solicitation from humans. Minimizing query complexity is hence important.

From a combinatorial optimization point of interest, MFAST is NP-Hard [4] but admits a PTAS [10] (see references therein for a more elaborate history of this important problem). The problem also has a machine learning theoretical interpretation, if we think of the directionality of the edge connecting u and v in the tournament T as a label. An ERM result has been obtained by Ailon [1] very recently. Interestingly, although Ailon's algorithm is not computationally efficient, it relies quite heavily on the ideas used in the PTAS [10].

In this work we obtain requirements (R1),(R2) and (R3) simultaneously, for both k -CC and MFAST.

2 Notations

For a natural number n we denote by $[n]$ the set of integers $\{1, \dots, n\}$. Let V denote a ground set of n elements. In the k -CC problem, V is endowed with an undirected graph $G = (V, E)$. A solution to the problem is given as a clustering $\mathcal{C} = \{C_1, \dots, C_k\}$ of V into k disjoint parts. We define $\equiv_{\mathcal{C}}$ to be the equivalence relation in which C_1, \dots, C_k are the equivalence classes. Equivalently, we view a solution as an undirected graph $G(\mathcal{C}) = (V, E(\mathcal{C}))$ in which $(u, v) \in E(\mathcal{C})$ if and only if $u \equiv_{\mathcal{C}} v$. The cost $\text{cost}_G(\mathcal{C})$ of a solution \mathcal{C} is the cardinality of the symmetric difference between the sets E and $E(\mathcal{C})$. When the input is clear from the context, we will simply write $\text{cost}(\mathcal{C})$.

In the MFAST problem, V is endowed with a tournament graph $T = (V, A)$.⁴ A solution is an injective function $\pi : V \mapsto [n]$ (a permutation). We define \prec_{π} to denote the induced order relation, namely: $u \prec_{\pi} v$ if and only if $\pi(u) < \pi(v)$. Equivalently, a solution can be viewed as a tournament $T(\pi) = (V, A(\pi))$, where $(u, v) \in A(\pi)$ if and only if $u \prec_{\pi} v$. The loss $\text{cost}_T(\pi)$ of a solution is the number of edges $(u, v) \in T(\pi)$ such that $(v, u) \in A$. In words, a unit cost is incurred for each inverted edge. When the input T is clear from the context, we may simply write $\text{cost}(\pi)$.

3 Statement of Results and Method Overview

As in [9,10], our query efficient PTAS for both k -CC and MFAST, distinguishes between a high cost case and a low cost case. In MFAST, *high cost* means that the optimal solution has cost at least $P(\varepsilon)n^2$, where $P(\varepsilon) = \Theta(\varepsilon^2)$. In k -CC, high cost means that the optimal solution has cost at least $Q(\varepsilon, k)n^2$, where $Q(\varepsilon, k) = \Theta(\varepsilon^6/k^{18})$.

In the low cost case, the problem has been solved for MFAST by Ailon [1]. There it is shown that $O(n\varepsilon^{-4} \log^4 n)$ edges from T are sufficient for finding a $(1 + \varepsilon)$ -approximate solution, in $\text{poly}(n, \varepsilon^{-1})$ time. We refer the reader to [1] for the details. As for the low cost case for k -CC, we show a PTAS with $o(n^2)$ query complexity in Section 4. The main idea of the algorithm is similar to that in

⁴ A *tournament* means that exactly one of (u, v) or (v, u) are in A for all $u \neq v$.

[9], but defers in a significant way. Roughly speaking, both algorithms choose a sample of vertices and enumerate over k -clusterings of the sample, while trying to compute optimal big clusters from the sample. In [9], for each such choice of sample k -clustering, a clustering of V is chosen, and recursion is executed on the union of small clusters. Here, we use the sample *in vitro* to learn a strong structural property of any optimal solution for the entire input. In particular, we don't need to return from a recursion to perform this learning.

For the high cost case we invoke an algorithm giving an additive $\varepsilon P(\varepsilon)n^2$ (resp. $\varepsilon Q(\varepsilon, k)n^2$) approximation for MFAST (resp. for k -CC). To that end, we use a standard LP based technique [5], together with another double sampling trick necessary for query efficiency, which we describe in Section 5 for the MFAST case (the k -CC case is easier). The main result there is as follows:

Theorem 1. *There exists a polynomial (in n) time algorithm for obtaining an additive $\varepsilon P(\varepsilon)n^2$ (resp. $\varepsilon Q(\varepsilon, k)n^2$) approximation for MFAST (resp. for k -CC). The algorithm queries $O(\varepsilon^{-2}P^{-2}(\varepsilon)n \log n)$ (resp. $O(\varepsilon^{-2}Q^{-2}(\varepsilon, k)n^2)$) input edges and runs in time $n^{O(\varepsilon^{-2}P^{-2}(\varepsilon) \log P(\varepsilon))}$ (resp. $n^{O(\varepsilon^{-2}Q^{-2}(\varepsilon, k) \log k)}$).*

In order to know whether we are at all in the high cost case, we apply the additive approximation algorithm in any case, and approximate the cost of the returned solution to within an additive error of $\Theta(P(\varepsilon)n^2)$ (resp. $\Theta(Q(\varepsilon, k)n^2)$). This estimation can clearly be done, with success probability at least $1 - n^{-10}$, by sampling at most $O(P^{-2}(\varepsilon) \log n)$ (resp. $O(Q^{-2}(\varepsilon, k) \log n)$) edges, by standard measure concentration arguments.⁵ This bound is overwhelmed by the bounds of Theorem 1. Our main results are summarized as follows.

Theorem 2. *There exists a PTAS for k -CC running in time $n^{O(\varepsilon^{-14}k^{36} \log k)}$ and requiring at most $O(\varepsilon^{-14}k^{36}n \log n)$ edge queries. With probability at least $1 - n^{-3}$, it outputs a clustering \tilde{C} with $\text{cost}(\tilde{C}) \leq \text{cost}(C^*)(1 + \varepsilon)$, where C^* is an optimal solution.*

Theorem 3. *There exists a PTAS for MFAST running in time $n^{O(\varepsilon^{-6})}$ and requiring at most $O(\varepsilon^{-6}n \log n + \varepsilon^{-4}n \log^4 n)$ edge queries. With probability at least $1 - n^{-3}$, it outputs a permutation σ with $\text{cost}(\sigma) \leq \text{cost}(\pi^*)(1 + \varepsilon)$, where π^* is an optimal solution.*

Note that the running times are overwhelmed by the high cost case in both, and the query complexity is overwhelmed by the high cost case in Theorem 2. We also note that we did not make a real effort to optimize the constants, including the exponents of k, ε .

4 Query Efficient PTAS for Low Cost in k -CC

We study the low cost case of k -CC on input $G = (V, E)$, and analyze an algorithm satisfying (R1)+(R2)+(R3). We need two ingredients. In Section 4.1

⁵ Note that the algorithm in [1] relies on a divide and conquer recursive strategy, in which the high cost algorithm and test must be implemented at each recursion node. This also holds for our k -CC algorithm, which identifies large clusters and then recurses on small ones. The recursive calls must solve and test for the high cost case as well.

we approximate the contribution of a single node v to the cost of any solution identical to the optimal solution except (maybe) for a change in the cluster to which v belongs. In Section 4.2 we achieve the PTAS, using a strategy similar to that of Giotis et al. in [9]: Identification of the large clusters in the optimal solution and recursion on the remainder. Note that the algorithm of [9] does not satisfy (R3), hence ours makes better use of the queried information.

4.1 An additive approximation of vertex costs

A major component in our PTAS for k -clustering is an additive approximation for the contribution of each vertex to the cost of the clustering. We start by formally defining this contribution, and then present Algorithm 1 and its analysis.

Definition 1. Let $\mathcal{C}^* = \{C_1^*, \dots, C_k^*\}$ be an optimal k -clustering, and assume its cost is γn^2 for some $\gamma \geq 0$. For $v \in V$ let $j^*(v)$ be defined as the unique index such that $v \in C_{j^*(v)}^*$. Let $C^*(v) = C_{j^*(v)}^*$. Let $\mathbf{1}_{u+v}$ be an indicator variable for the predicate $(u, v) \in E$, and similarly define the complement $\mathbf{1}_{u-v} = 1 - \mathbf{1}_{u+v}$. Let $\deg_+(v, j) = \sum_{u \in C_j^* \setminus \{v\}} \mathbf{1}_{u+v}$, $\deg_-(v, j) = \sum_{u \in C_j^* \setminus \{v\}} \mathbf{1}_{u-v}$, $\degout_+(v, j) = \sum_{u \in (V \setminus C_j^*) \setminus \{v\}} \mathbf{1}_{u+v}$, and $\degout_-(v, j) = \sum_{u \in (V \setminus C_j^*) \setminus \{v\}} \mathbf{1}_{u-v}$. Let $\text{cost}^*(v) = \sum_{u \in C^*(v) \setminus \{v\}} \mathbf{1}_{u-v} + \sum_{u \notin C^*(v)} \mathbf{1}_{u+v}$. Notice that $\text{cost}^*(v) = \deg_-(v, j(v)) + \degout_+(v, j(v))$ and $\text{cost}(\mathcal{C}^*) = \frac{1}{2} \sum_v \text{cost}^*(v)$. For any $j \in [k]$, let $\text{cost}^*(v, j) \triangleq \deg_-(v, j) + \degout_+(v, j)$. That is, $\text{cost}^*(v, j)$ is the contribution of the vertex v to the cost of the clustering that is identical to \mathcal{C}^* , except the location of v , which is reset to C_j^* .

Algorithm 1 Additive approximation of cost^*

Input: A graph $G = (V, E)$, a parameter $\beta > 0$ and integer $k > 1$

Output: $\forall v \in V$, and $j \in [k]$, an estimation $\widetilde{\text{cost}}(v, j)$ to $\text{cost}(v, j)$

Choose $S = (v_1, \dots, v_t)$, where $t = c \log(n) \beta^{-9}$ (c is some sufficiently large universal constant), be a multiset of i.i.d. uniformly randomly chosen vertices from V . Let $\tilde{S}_1, \dots, \tilde{S}_k$ be an optimal k -clustering for the reduced problem $(S, E|_S)$ (where $E|_S = E \cap (S \times S)$), where the solution is found using exhaustive search.

For any $v \in V$, $j \in [k]$, let: $\widetilde{\deg}_-(v, j) \triangleq \sum_{v \in \tilde{S}_j \setminus \{v\}} \mathbf{1}_{u-v}$ and $\widetilde{\degout}_+(v, j) \triangleq \sum_{v \in (S \setminus \tilde{S}_j) \setminus \{v\}} \mathbf{1}_{u+v}$. (The summations count elements of S with multiplicities.)

Output for every $v \in V$, $j \in [k]$ the estimation:

$$\widetilde{\text{cost}}(v, j) \triangleq \frac{n}{|S|} \left(\widetilde{\deg}_-(v, j) + \widetilde{\degout}_+(v, j) \right).$$

The rest of this section proves the following guarantee of Algorithm 1.

Theorem 4. Fix $\beta > 0$, to be passed as parameter to Algorithm 1. There exist some universal constant c such that if $\gamma \leq c\beta^6$ then for all $v \in V$, $j \in [k]$ it holds

that for the output of the algorithm, after possibly renaming the optimal clusters $\{C_1^*, \dots, C_k^*\}$, $\left| \widetilde{\text{cost}}(v, j) - \text{cost}^*(v, j) \right| < \beta n$. For any input, Algorithm 1 will run in $n^{O(\beta^{-9} \log k)}$ time and will require at most $O(n \log(n) \beta^{-9})$ edge queries.

The claim regarding the time and query complexity of the algorithm are trivial. Indeed, the time is dominated by exhaustively searching the space of k -clusterings of the sample S in the algorithm. We focus on proving the correctness. We need some more definitions.

Definition 2. Let $u, v \in V$, S a multi-subset of V , $j \in [k]$ and $\delta > 0$. Let $\deg_+^S(v, j) = \sum_{u \in (C_j^* \cap S) \setminus \{v\}} \mathbf{1}_{u+v}$, $\deg_-^S(v, j) = \sum_{u \in (C_j^* \cap S) \setminus \{v\}} \mathbf{1}_{u-v}$, $\text{degout}_+^S(v, j) = \sum_{u \in (S \setminus C_j^*) \setminus \{v\}} \mathbf{1}_{u+v}$ and $\text{degout}_-^S(v, j) = \sum_{u \in (S \setminus C_j^*) \setminus \{v\}} \mathbf{1}_{u-v}$, where the summations take multiplicities in S into account. Let $\text{cost}^{*S}(v) \triangleq \deg_-^S(v, j^*(v)) + \text{degout}_+^S(v, j^*(v))$.

In what follows, set $\delta = \Theta(\beta^3)$. Define \mathcal{S} as the partition of S (from Algorithm 1) induced by C^* . That is $\mathcal{S} = \{S_1, \dots, S_k\}$ where $S_j = C_j^* \cap S$.

Lemma 1. With probability at least $1 - n^{-10}$, for all $v \in V$ and $j \in [k]$,

$$\max\{ |\deg_+^S(v, j)/n - \deg_+^S(v, j)/|S||, |\deg_-^S(v, j)/n - \deg_-^S(v, j)/|S||, \\ |\text{degout}_+^S(v, j)/n - \text{degout}_+^S(v, j)/|S||, |\text{degout}_-^S(v, j)/n - \text{degout}_-^S(v, j)/|S|| \} = O(\delta) .$$

The simple proof is deferred to Appendix A. From Lemma 1,

Lemma 2. Assume $\gamma = o(\delta)$. With probability $1 - n^{-10}$, the cost of the partition \mathcal{S} on the graph $G|_S = (S, E|_S)$ is at most $O(\delta|S|^2)$.

In the following lemma we show that any pair of clusterings that are close w.r.t. to their edges are also close w.r.t. their vertices.

Lemma 3. Let $\mathcal{S}, \tilde{\mathcal{S}}$ be two k -clusterings of S , and let $E(\mathcal{S}), E(\tilde{\mathcal{S}})$ be their corresponding edge sets, namely, $(u, v) \in E(\mathcal{S})$ if and only if $u \equiv_{\mathcal{S}} v$, and similarly for $\tilde{\mathcal{S}}$. Assume the size of the symmetric difference between $E(\mathcal{S})$ and $E(\tilde{\mathcal{S}})$ is at most $\delta|S|^2$, where $\delta < c/k^3$ and c is a sufficiently small constant. Then for some reordering of indices, for every $j \in [k]$, $\max\{|S_j \setminus \tilde{S}_j|, |\tilde{S}_j \setminus S_j|\} = O(\delta^{1/3}|S|)$.

We will only present a main structural claim used by the proof. The remainder of the proof will be deferred to Appendix C.

Proof. We start with an auxiliary claim showing that every cluster in \mathcal{S} has a similar cluster in $\tilde{\mathcal{S}}$ (and vice versa).

Claim. Let C be a cluster of \mathcal{S} . There exists some cluster D in $\tilde{\mathcal{S}}$ such that $|C \setminus D| \leq O(\delta^{1/3}|S|)$.

Proof. Let D be a cluster in \tilde{S} that maximizes $|D \cap C|$. Let $A = D \cap C$ and let $\bar{A} = C \setminus A$. Notice that for every pair $(u, v) \in A \times \bar{A}$ the edge (u, v) is an element of $E(\mathcal{S}) \setminus E(\tilde{\mathcal{S}})$. Hence, $|A||\bar{A}| \leq \delta|S|^2$. If $|C| < \delta^{1/3}|S|$ then the claim holds trivially. If $|C| \geq \delta^{1/3}|S|$, then we get: $|A|(|C| - |A|) = |A||\bar{A}| \leq \delta|S|^2 \leq \delta^{1/3}|C|^2$. A simple calculation will show that either $|A| \leq O(\delta^{1/3}|C|)$ or $|A| \geq |C|(1 - O(\delta^{1/3}))$. By setting the constant c to be sufficiently small we get that the first option implies $|A| < |C|/k$ which is impossible due to the fact that A maximizes $|D \cap C|$ over all clusters D in \tilde{S} . definition of A . We conclude that $|C \setminus D| = O(\delta^{1/3}|S|)$, proving the claim. The remainder of the proof of Lemma 3 is deferred to Appendix C.

Let $\tilde{S} = \tilde{S}_1, \dots, \tilde{S}_k$ be an optimal k -clustering of the induced input $G|_S$. By Lemma 2, we know that with probability at least $1 - n^{-10}$ the cost of the solution \tilde{S} is at most $\delta|S|^2$. By the triangle inequality, this implies that the symmetric difference between $E(\mathcal{S})$ and $E(\tilde{\mathcal{S}})$ is at most $O(\delta|S|^2)$. Hence, we may apply Lemma 3 and assume that the clusters S_1, \dots, S_k and $\tilde{S}_1, \dots, \tilde{S}_k$ are aligned with each other. Define: $\widetilde{\deg}_+(v, j) = \sum_{u \in \tilde{S}_j \setminus \{v\}} \mathbf{1}_{u+v}$, $\widetilde{\deg}_-(v, j) = \sum_{u \in \tilde{S}_j \setminus \{v\}} \mathbf{1}_{u-v}$, $\widetilde{\deg}_{out+}(v, j) = \sum_{u \in (S \setminus \tilde{S}_j) \setminus \{v\}} \mathbf{1}_{u+v}$, and $\widetilde{\deg}_{out-}(v, j) = \sum_{u \in (S \setminus \tilde{S}_j) \setminus \{v\}} \mathbf{1}_{u-v}$.

Lemma 4. *With probability at least $1 - n^{-8}$, for all $v \in V$ and $j \in [k]$, $\left| \frac{\widetilde{\deg}_+(v, j)}{|S|} - \frac{\deg_+(v, j)}{n} \right| = O(\delta^{1/3})$. The same is true for the other ‘deg functions’.*

Proof. By the guarantee of Lemma 3, for all $v \in V, j \in [k]$, $\left| \frac{\widetilde{\deg}_+(v, j)}{|S|} - \frac{\deg_+^S(v, j)}{|S|} \right| = O(\delta^{1/3})$. By the guarantee of Lemma 1, we have that for all $v \in V, j \in [k]$, $\left| \frac{\deg_+^S(v, j)}{|S|} - \frac{\deg_+(v, j)}{n} \right| = O(\delta^{1/3})$. The claim follows by union bounding and using the triangle inequality. This concludes the lemma’s proof.

Theorem 4 is now an easy corollary.

4.2 The PTAS

In this section we utilize the approximations to the costs of the vertices achieved in Algorithm 1 to achieve a PTAS for k -clustering. We note that the heart of our contribution is the previous section, and the lemmas and proofs here follow the lines of [9]. The main algorithm (Algorithm 4.2) is of course different since it utilizes the results of the previous section.

Throughout this section we will assume that the optimal clustering \mathcal{C}^* has a cost of γn^2 where $\gamma < c_1 \beta^6$, where the parameter β will be taken as $c_2 \varepsilon / k^2$, and c_1, c_2 will be sufficiently small constants so that Theorem 4 is satisfied.

The remainder of the section is dedicated to proving Theorem 2. We need some lemmas. In what follows, we assume that the invocation of Algorithm 1 is successful in the sense that the guarantee of Theorem 4 holds. The following is an immediate corollary of this guarantee.

Lemma 5. *Let $v \in V$ be a vertex satisfying $v \in \hat{C}_j \cap C_i^*$, where $i \neq j$. Then $\text{cost}^*(v, j) \leq \text{cost}^*(v) + 2\beta n$.*

Algorithm 2 PTAS for k -CC (low cost)

Input: A graph $G = (V, E)$, an integer $k > 1$ and a parameter $\varepsilon > 0$. It is assumed that the optimal k -CC cost of G is γn^2 , where $\gamma < c_1 \beta^6$ and $\beta = c_2 \varepsilon / k^3$.

Output: A clustering $\tilde{\mathcal{C}} = \{\tilde{C}_1, \dots, \tilde{C}_k\}$ of G .

Run Algorithm 1 with inputs G, k and β . Obtain approximations $\widetilde{\text{cost}}(v, j)$ for all $v \in V$ and $j \in [k]$.

Create empty clusters $\hat{C}_1, \dots, \hat{C}_k$. For all $v \in V$ add v to \hat{C}_i , where $i = \arg\min_j \{\text{cost}(v, j)\}$.

Reorder the clusters so that $|\hat{C}_1| \geq \dots \geq |\hat{C}_k|$. Let $\ell \in [k]$ be such that $|\hat{C}_\ell| \geq \frac{n}{2k}$ and $|\hat{C}_{\ell+1}| < \frac{n}{2k}$ (if no such integer exists, set $\ell = k$).

Run the algorithm recursively on the restriction of G on $W \triangleq \cup_{j>\ell} \hat{C}_j$, the integer $k - \ell$ and approximation parameter $\varepsilon(1 - 1/k)$. Denote its output by $\tilde{C}_{\ell+1}, \dots, \tilde{C}_k$.

Output $\tilde{\mathcal{C}} = (\tilde{C}_1 = \hat{C}_1, \dots, \tilde{C}_\ell = \hat{C}_\ell, \tilde{C}_{\ell+1}, \dots, \tilde{C}_k)$.

Define for any $v \in V$, $\widetilde{\text{cost}}(v) = \min_{j \in [k]} \widetilde{\text{cost}}(v, j)$, where $\widetilde{\text{cost}}(v, j)$ is as defined in Algorithm 2. Define $V_{\text{costly}} = \{v \in V \mid \widetilde{\text{cost}}(v) \geq c_3 n / k^2\}$, where c_3 is some sufficiently small constant. For any $v \in V_{\text{costly}}$, $\text{cost}^*(v) \geq \frac{1}{2} c_3 n / k^2$ due to the guarantee of Theorem 4 and our choice of β . Since (twice) the total optimal cost is bounded by that incurred by vertices in V_{costly} :

$$|V_{\text{costly}}| \leq 4\gamma n^2 / (c_3 n / k^2) \leq (4\gamma n k^2) / c_3. \quad (1)$$

In particular, using a very crude estimate, this means

$$|V_{\text{costly}}| \leq c_4 n / k, \quad (2)$$

where c_4 is a constant that can be made sufficiently small by reducing c_1 as necessary. Recall that $\hat{C}_1, \dots, \hat{C}_\ell$ are the large clusters found by Algorithm 2. Notice that since there are k clusters, there must be at least one cluster of size $\geq \frac{n}{2k}$ meaning that $\ell \geq 1$.

Lemma 6. *For any $j \in [\ell]$, $C_j^* \setminus V_{\text{costly}} = \hat{C}_j \setminus V_{\text{costly}}$.*

The proof is deferred to Appendix B for lack of space. The next lemma states the existence of a clustering whose large clusters are identical to those found by our algorithm and has an almost optimal cost.

Lemma 7. *There exist some k -clustering of V , $\mathcal{D} = (D_1, \dots, D_k)$ such that for all $j \in [\ell]$, $D_j = \hat{C}_j$ and $\text{cost}(\mathcal{D}) \leq \gamma n^2(1 + \varepsilon/k)$*

Proof. Take \mathcal{D} to be the clustering defined as follows. For any $i \in [k]$, $D_i = (C_i^* \setminus V_{\text{costly}}) \cup (\hat{C}_i \cap V_{\text{costly}})$. That is, D_i is the result starting with the clustering \mathcal{C} and of moving the vertices of V_{costly} to the clusters according to $\hat{\mathcal{C}} = \{\hat{C}_1, \dots, \hat{C}_k\}$.

Denote by $\text{cost}^{\mathcal{D}}(v)$ the cost of a vertex v w.r.t. the partition \mathcal{D} . Notice that the only edges for which the clustering \mathcal{D} pays for while the clustering \mathcal{C}^* does not must be incident to a node in V_{costly} . Hence,

$$\text{cost}(\mathcal{D}) - \text{cost}(\mathcal{C}^*) \leq \sum_{v \in V_{\text{costly}}} (\text{cost}^{\mathcal{D}}(v) - \text{cost}^*(v)). \quad (3)$$

Assume $v_{\text{costly}} \in V_{\text{costly}} \cap D_j$ for some $j \in [k]$. Clearly

$$|\text{cost}^{\mathcal{D}}(v_{\text{costly}}) - \text{cost}^*(v_{\text{costly}}, j)| \leq |V_{\text{costly}}|, \quad (4)$$

because the only difference in such a vertex's cost can come from edges connecting it to other vertices in V_{costly} . Now assume $v_{\text{costly}} \in V_{\text{costly}} \cap C_i^* \cap D_j$ for $j \neq i$. By construction, $v_{\text{costly}} \in \hat{C}_j$. By Lemma 5, this implies $\text{cost}^*(v_{\text{costly}}, j) \leq \text{cost}^*(v_{\text{costly}}) + 2\beta$. By (4) we conclude

$$\begin{aligned} \text{cost}^{\mathcal{D}}(v_{\text{costly}}) - \text{cost}^*(v_{\text{costly}}) &\leq \\ (\text{cost}^*(v_{\text{costly}}, j) - \text{cost}^*(v_{\text{costly}}, i)) + |V_{\text{costly}}| &\leq 2\beta + |V_{\text{costly}}|. \end{aligned}$$

Plugging this into (3) and using (1), we get

$$\text{cost}(\mathcal{D}) - \text{cost}(\mathcal{C}^*) \leq |V_{\text{costly}}| (2\beta n + |V_{\text{costly}}|) \leq \gamma n^2 (8\beta k^2 / (c_3) + 16\gamma k^4 / (c_3)^2).$$

The claim follows since $\gamma < c_1 \beta^6 \leq \frac{\varepsilon}{2k} \cdot \frac{(c_3)^2}{16k^4}$, $\beta < \frac{\varepsilon}{2k} \cdot \frac{c_3}{8k^2}$ assuming small c_1, c_2 .

Proof (of Theorem 2). The claim regarding the query complexity is trivial given Theorem 4. The running time is a result of the recursion formula $T(n, \varepsilon, k) = n^{\varepsilon^{-9} k^{27} \log(k)} + T(n, \varepsilon(1 - 1/k), k - 1) = n^{\varepsilon^{-9} k^{27} \log(k)}$. We note that in [9], the stated running time is doubly exponential in k whereas here it is singly exponential in k . This difference is due to a minor observation that the recursive call should be with the parameter $\varepsilon(1 - 1/k)$ rather than $\varepsilon/10$. The same minor change would result in a singly exponential dependence in k in the algorithm given in [9] as well. Let W be the union of the small clusters. That is, $W = \cup_{j=\ell+1}^k \hat{C}_j$. By lemma 7, all of the vertex pairs that are not contained in the set $W \times W$ incur a cost in \hat{C} identical to that in \mathcal{D} . Let d_1 be the cost of \mathcal{D} on pairs in $W \times W$ and let d_2 be its cost on the remaining pairs $V \times V \setminus W \times W$. Since W is clustered recursively, we have that the cost of \hat{C} is at most $d_2 + d_1(1 + \varepsilon/k) \leq (d_1 + d_2)(1 + \varepsilon/k) = \text{cost}(\mathcal{D})(1 + \varepsilon/k)$. The statement of the theorem follows.

5 Query Efficient PTAS for High Cost

We present a query efficient PTAS for the high loss case of MFAST. The query efficient PTAS for the high loss case of k -CC is almost identical and is thus not presented. We will start by describing a known PTAS ((R1)+(R2)) based on an approach given by Arora et. al. We then show how to add requirement (R3). The final approach is summarized in Algorithm 3, found in Appendix D.

5.1 (R1)+(R2) using a Known Additive Approximation Algorithm

Let π^* denote an optimal permutation, and let $\text{OPT} = \text{cost}(\pi^*) = \text{cost}(\pi^*)$. In the high cost MFAST case, as explained in Section 3, we assume $\text{OPT} \geq \gamma n^2$, where $\gamma = \Theta(\varepsilon^2)$. Instead of directly solving MFAST, we solve the *bucketed* version. This idea is not new and can be found in e.g. [10]. An m -bucket ordering σ of V is a mapping $\sigma : V \mapsto [m]$, where for each $i \in [m]$ the preimage satisfies: $\frac{n}{2m} \leq |\sigma^{-1}(i)| \leq \frac{2n}{m}$. For brevity we say that $u <_{\sigma} v$ if $\sigma(u) < \sigma(v)$, and $u \equiv_{\sigma} v$ if $\sigma(u) = \sigma(v)$. We extend the definition of $\text{cost}(\cdot)$ to bucketed orders by defining $\text{cost}(\sigma) \triangleq \sum_{u <_{\sigma} v} \mathbf{1}_{(v,u) \in A}$. We will also need to define:

$\text{cost}^{u,v}(\sigma) \triangleq \mathbf{1}_{u <_\sigma v} \mathbf{1}_{(v,u) \in A} + \mathbf{1}_{v <_\sigma u} \mathbf{1}_{(u,v) \in A}$ and $\text{cost}^u(\sigma) \triangleq \frac{1}{2} \sum_{v \in V} \text{cost}^{u,v}(\sigma)$, so that $\text{cost}(\sigma) = \sum_{u \in V} \text{cost}_T^u(\sigma)$. A permutation π extends an m -bucketed ordering σ if $u <_\pi v$ whenever $u <_\sigma v$.

Observation 5 [10] *For any π extending σ , $|\text{cost}(\pi) - \text{cost}(\sigma)| = O(n^2/m)$, hence for the purpose of obtaining a $(1 + \varepsilon)$ -approximate solution in our case it suffices to consider m -bucketed orderings with $m = \Theta(1/(\varepsilon\gamma))$.*

Let σ^* denote any m -bucketed ordering of V of which π^* is an extension, and such that $\lfloor n/m \rfloor \leq |(\sigma^*)^{-1}(i)| \leq \lceil n/m \rceil$ for all $i \in [m]$. The following approach has been taken in [5]. Let $S = (v_1, v_2, \dots, v_s)$ be a random series of $s = O(\log n / (\varepsilon\gamma)^2)$ vertices in V , each element chosen uniformly and independently, with repetitions. Abusing notation, we will also think of S as the series $\{v_1, \dots, v_s\}$. For each m -bucketed ordering σ and for each $u \in V$, we make the following definitions: $\text{cost}^{u,S}(\sigma) \triangleq \frac{n}{2s} \sum_{i=1}^s \text{cost}^{u,v_i}(\sigma)$ and $\text{cost}^S(\sigma) \triangleq \sum_{u \in V} \text{cost}^{u,S}(\sigma)$. Clearly $\text{cost}^S(\sigma)$ is an unbiased estimator of $\text{cost}(\sigma)$ over the choice of the sample S . The top level of our algorithm will enumerate over all $n^{\Theta(\log(1/(\varepsilon\gamma))/(\varepsilon\gamma)^2)}$ possibilities for the value of $(\sigma^*(v_1), \dots, \sigma^*(v_s))$. From now on, we will assume the correct possibility has been chosen, so that $\sigma^*(v)$ is “known” for $v \in S$. A verification step will be used to identify the correct possibility in the end (see Algorithm 3 in Appendix D).

Definition 3. *For an m -bucket ordering σ , a vertex $u \in V$ and integer $i \in [m]$, let $\sigma_{u \rightarrow i}$ denote the bucket order defined by leaving the value of $\sigma(v)$ unchanged for $v \neq u$ and mapping u to i . More precisely: $\sigma_{u \rightarrow i}(v) = \sigma(v)$ if $v \neq u$, and $\sigma_{u \rightarrow i}(u) = i$.*

Note that $\sigma_{u \rightarrow i}$ may not be exactly an m -bucket ordering. To be precise, we will say that $\sigma_{u \rightarrow i}$ is an m -bucket* ordering whenever σ is an m -bucket ordering, for every $u \in V$ and $i \in [m]$. Clearly, Observation 5 holds for m -bucket* orderings as well, with a possible different constant hiding in the Θ -notation. The following lemma is proven using standard measure concentration inequalities:

Lemma 8. *Fix an m -bucket ordering σ of V . With probability at least $1 - n^{-10}$, for all $u \in V$ and $i \in [m]$:*

$$|\text{cost}^{u,S}(\sigma_{u \rightarrow i}) - \text{cost}^u(\sigma_{u \rightarrow i})| = O(\varepsilon\gamma n). \quad (5)$$

By summing (5) over all (u, i) such that $i = \sigma(u)$, we get

Corollary 1. *For any m -bucket order σ with probability at least $1 - n^{-10}$, $|\text{cost}^S(\sigma) - \text{cost}(\sigma)| = O(\varepsilon\gamma n^2)$.*

Arora et al’s LP approach [5] The benefit of Lemma 8 is the fact that (5) can be written as a pair of linear inequalities in variables $(x_{vj})_{v \in V \setminus \{u\}, j \in [m]}$, where x_{vj} is indicator for the predicate “ $\sigma(v) = j$ ”. Indeed, $\text{cost}^{u,S}(\sigma_{u \rightarrow i})$ is a known constant, and $\text{cost}^u(\sigma_{u \rightarrow i})$ is a linear combination of $(x_{vj})_{v \neq u, j \in [m]}$. This property allowed Arora et. al in [5] to introduce an LP over these variables,

where the utility function $\text{cost}^S(\sigma)$ is clearly a linear function of the system $(x_{vj})_{v \in V, j \in [m]}$. Some obvious standard constraints are added: For all v, j , $x_{vj} \geq 0$ and for all v , $\sum_{j \in [m]} x_{vj} = 1$, and of course x_{vj} is hardwired as 1 (resp. 0) whenever $v \in S$ and $\sigma^*(v) = j$ (resp. $\sigma^*(v) \neq j$). The *almost balanced bucket* constraint is also added: $\forall j \in [m] : \lfloor n/m \rfloor \leq \sum_{v \in V} x_{vj} \leq \lceil n/m \rceil$. The following arguments in [5] are by now classic: Randomly round the optimal LP solution by independently drawing, for each $v \in V$, from the discrete distribution assigning probability x_{vj}^* to the j 'th bucket. Denote the resulting m -bucket order σ' . As argued in [5], with high probability each constraint in the system will be satisfied up to a possible additive violation of magnitude depending on an ℓ_∞ and an ℓ_0 (support size) property of the constraint. The precise statement is as follows:

Lemma 9 (Essentially [5]). *If the optimal solution to the LP x^* satisfies $\sum \beta_{vj} x_{vj}^* \leq \alpha$ for $\beta \in \mathbf{R}^{|V| \times m}$ and $\alpha \in \mathbf{R}$, then with probability at least $1 - \eta$ the rounded solution σ' will violate the constraint by no more than $\|\beta\|_\infty \sqrt{\|\beta\|_0 \log(1/\eta)}$, where $\|\beta\|_0$ is the number of vertices $v \in V$ such that $\beta_{vi} \neq 0$ for some $i \in [m]$, $\|\beta\|_\infty = \max_{v \in V, i \in [m]} |\beta_{vi}|$ and $\eta > 0$ is any number.*⁶

In our case, consider an LP constraint coming from (8). Its corresponding coefficient vector β satisfies $\|\beta\|_0 \leq n$ and $\|\beta\|_\infty \leq 1$. We conclude that with probability at least $1 - n^{-10}$, (5) is satisfied with $\sigma = \sigma'$ and for all u, i , and hence also the guarantee of Corollary 1.⁷ Also, in virtue of the *almost balanced bucket* constraint and Lemma 9, with probability at least $1 - n^{-10}$ the rounded solution σ' is an m -bucket order. Additionally, by analyzing the coefficient vector β_{utility} corresponding to the LP utility function, with probability at least $1 - n^{-10}$ the cost $\text{cost}^S(\sigma')$ is bounded by $\text{LP}(x^*) + O(\varepsilon \gamma n^2)$, which is bounded by $\text{cost}^S(\sigma^*) + O(\varepsilon \gamma n^2)$ by LP optimality. Note also that the guarantee of Corollary 1 also applies to $\sigma = \sigma^*$ with probability at least $1 - n^{-10}$. Combining using union bound and triangle inequality, one gets that $\text{cost}(\sigma') \leq \text{cost}(\sigma^*) + O(\varepsilon \gamma n^2)$. We conclude the section with the following lemma that is implicit in [5]

Lemma 10. *Given the correct bucketing on the vertices of S , one can construct a polynomially sized linear program whose rounded solution σ' has the property $\text{cost}(\sigma') \leq \text{cost}(\sigma^*) + O(\gamma \varepsilon n^2)$ with probability at least $1 - n^{-9}$.*

5.2 Query efficiency

The problem is that expressing inequality (8) in the LP requires complete knowledge of the input T . If we take a revised look at this strategy, we see that the sample S is not strong enough in the sense that it can be used to well approximate $\text{cost}(\sigma)$ (per Corollary 1) for no more than $\text{poly}(n)$ m -bucket orders σ simultaneously, but certainly not for *all* m -bucket orders.

For each $u \in V$ randomly select a sample $S^u = (v_1^u, \dots, v_p^u)$ of vertices of V , where $p = O((\gamma \varepsilon)^{-2} \log n)$, each sample S^u is chosen independently of the other

⁶ We have implicitly viewed σ' as a vector $(\sigma'_{vj})_{v \in V, j \in [m]}$, with σ'_{vi} indicator for $\sigma'(v) = i$.

⁷ All this happens with with possibly slightly worse constants hiding in the O -notation.

samples, and the v_i^u 's are chosen uniformly at random from V , with repetitions. Denote the ensemble $\{S^u : u \in V\}$ by \mathcal{S} . For any m -balanced ordering π on V , define $\text{cost}^{u,\mathcal{S}}(\pi) = \frac{n}{2p} \sum_{i=1}^p \text{cost}^{u,v_i^u}(\pi)$ and $\text{cost}^{\mathcal{S}}(\pi) = \sum_{u \in V} \text{cost}^{u,\mathcal{S}}(\pi)$. It is not hard to see that $\text{cost}^{\mathcal{S}}(\pi)$ is an unbiased estimator of $\text{cost}(\pi)$, for any π . Using standard measure concentration bounds, we have the following:

Lemma 11. *With probability at least $1 - n^{-10}$, uniformly for all m -balanced orderings σ on V , $|\text{cost}^{\mathcal{S}}(\sigma) - \text{cost}(\sigma)| = O(\varepsilon \gamma n^2)$.*

Lemma 12. *Fix an m -balanced ordering σ . With probability at least $1 - n^{-10}$, uniformly for all $u \in V$ and $i \in [m]$,*

$$|\text{cost}^{u,\mathcal{S}}(\sigma_{u \rightarrow i}) - \text{cost}^{u,\mathcal{S}}(\sigma_{u \rightarrow i})| = O(\varepsilon \gamma n) . \quad (6)$$

By summing (6) over all (u, i) s.t. $i = \sigma(u)$, we get

Corollary 2. *Fix an m -balanced ordering σ . With probability at least $1 - n^{-10}$, $|\text{cost}^{\mathcal{S}}(\sigma) - \text{cost}(\sigma)| = O(\varepsilon \gamma n^2)$.*

We build an LP as in Section 5.1, except that (6) replaces (5). Note that the coefficient vectors β of the new constraints now satisfy $\|\beta\|_0 = O(p) = O((\gamma\varepsilon)^{-2} \log n)$ and $\|\beta\|_\infty = O(n/p) = O(n\gamma^2\varepsilon^2/\log n)$. Using Lemma 9 and a similar analysis as in Section 5.1, an analog of Lemma 10 can be proven. That is, we conclude that with probability at least $1 - n^{-9}$, the m -bucketed ordering σ' outputted by rounding the optimal LP solution satisfies $\text{cost}^{\mathcal{S}}(\sigma') \leq \text{cost}^{\mathcal{S}}(\sigma^*) + O(\varepsilon \gamma n^2)$. By Lemma 11 this implies that $\text{cost}(\sigma') \leq \text{cost}(\sigma^*) + O(\varepsilon \gamma n^2)$. Algorithm 3 (Appendix D) summarizes the query efficient PTAS for MFAST high cost case. The k -CC high cost case can be solved in similar lines, though this case is slightly easier because the clusters need not be balanced.

6 Discussion and Future Work

We believe that in the low cost k -CC case, there should be a PTAS with efficient query complexity, running in time $\text{poly}(n, \varepsilon^{-1}, k)$ (not exponential in k, ε^{-1}), assuming the low cost case in each recursive instance. This is true for MFAST, and we leave the question of achieving it for k -CC to future work.

References

1. Nir Ailon. An active learning algorithm for ranking from pairwise preferences with an almost optimal query complexity. *Journal of Machine Learning Research*, 13:137–164, 2012.
2. Nir Ailon and Ron Begleiter. Active learning of clustering with side information using smooth relative regret approximations. *arXiv:1201.6462*, 2012.
3. Nir Ailon, Moses Charikar, and Alantha Newman. Aggregating inconsistent information: Ranking and clustering. *Journal of the ACM*, 55(5):23:1–23:27, October 2008.
4. Noga Alon. Ranking tournaments. *SIAM Journal on Discrete Mathematics*, 20, 2006.

5. Sanjeev Arora, Alan M. Frieze, and Haim Kaplan. A new rounding procedure for the assignment problem with applications to dense graph arrangement problems. *Math. Program.*, 92(1):1–36, 2002.
6. Nikhil Bansal, Avrim Blum, and Shuchi Chawla. Correlation clustering. *Machine Learning*, 56:89–113, 2004.
7. Sugato Basu. *Semi-supervised Clustering: Probabilistic Models, Algorithms and Experiments*. PhD thesis, Department of Computer Sciences, University of Texas at Austin, 2005.
8. Moses Charikar and Anthony Wirth. Maximizing quadratic programs: Extending grothendieck’s inequality. In *FOCS*, pages 54–60. IEEE Computer Society, 2004.
9. Ioannis Giotis and Venkatesan Guruswami. Correlation clustering with a fixed number of clusters. *Theory of Computing*, 2(1):249–266, 2006.
10. C. Kenyon-Mathieu and W. Schudy. How to rank with few errors. In *STOC ’07: Proceedings of the thirty-ninth annual ACM Symposium on Theory of Computing*, pages 95–103. ACM, 2007.
11. Chaitanya Swamy. Correlation clustering: maximizing agreements via semidefinite programming. In *SODA*, pages 526–527, 2004.

A Proof of Lemma 1

This is a simple application of the following more general well known sampling principle: If V_1, \dots, V_M is a collection of subsets of V and T is a sample of N uniformly chosen elements from V (with repetition), then with probability $1 - \eta$, for all $i = 1, \dots, M$, $\left| \frac{|V_i|}{n} - \frac{|V_i \cap T|}{|T|} \right| = O\left(\sqrt{N^{-1} \log(M/\eta)}\right)$.

B Proof of Lemma 6

We start by proving the inclusion

$$\hat{C}_j \subseteq C_j^* \cup V_{\text{costly}}. \quad (7)$$

Assume for contradiction that there exist some $v \in \hat{C}_j \setminus (C_j^* \cup V_{\text{costly}})$. Let $i \in [k]$, $i \neq j$ be such that $v \in C_i^*$. As $v \notin V_{\text{costly}}$ we know by Theorem 4 that $\text{cost}^*(v, i) = \text{cost}^*(v) \leq \text{cost}^*(v, j) \leq \widetilde{\text{cost}}(v, j) + \beta n \leq \frac{c_3 n}{k^2} + \beta n$. We get:

$$\frac{2nc_3}{k^2} + 2n\beta \geq \text{cost}^*(v, i) + \text{cost}^*(v, j) \geq |C_i^*| + |C_j^*| - 1, \quad (8)$$

where the right hand inequality is a consequence of the fact for any $u \in (C_i^* \cup C_j^*) \setminus \{v\}$, v will either incur a price w.r.t. u if it is included in C_i^* or it is included in C_j^* . Hence,

$$\max\{|C_i^*|, |C_j^*|\} \leq \frac{2nc_3}{k^2} + 2n\beta + 1 \leq \frac{c_5 n}{k^2} \quad (9)$$

for some constant c_5 that can be made arbitrarily small by tuning c_2 (the constant product in β) and c_3 . Since this holds for any $i \neq j$ satisfying $C_i^* \cap (\hat{C}_j \setminus V_{\text{costly}}) \neq \emptyset$ we have that

$$|C_j^*| \geq |\hat{C}_j| - |V_{\text{costly}}| - \sum_{i: i \neq j, C_i^* \cap (\hat{C}_j \setminus V_{\text{costly}}) \neq \emptyset} |C_i^*| \geq \frac{n}{2k} - \frac{c_4 n}{k} - \frac{c_5 n}{k} > \frac{c_5 n}{k},$$

where we used (2) and (9), and ensure that $2c_5 + c_4 < 1/2$. We derive a contradiction to (9).

We now prove that the inclusion $C_j^* \subseteq \hat{C}_j \cup V_{\text{costly}}$. Notice that by (7), we conclude that $|C_j^*|$ is lower bounded by $\frac{n}{k} (\frac{1}{2} - c_4) \geq \frac{n}{4k}$, as long as $c_4 < 1/4$. Assume for the sake of contradiction that there exists some $v \in C_j^* \setminus V_{\text{costly}}$ such that for some $i \neq j$, $v \in \hat{C}_i$. By the guarantee of Theorem 4 we have that $\text{cost}^*(v, j) \leq \text{cost}^*(v, i) \leq \widetilde{\text{cost}}(v, i) + \beta n \leq c_3 n/k^2 + \beta n$. This gives us again (8), leading to (9), contradicting our lower bound on $|C_j^*|$ for sufficiently small c_5 . This concludes the lemma proof.

C Continuation of Proof of Lemma 3

We now proceed with the proof of the lemma.

Consider the following bipartite directed graph $H = (U, \Gamma)$. The vertex set U is defined as follows: $U = \{C : C \text{ is a cluster in } \mathcal{S}\} \cup \{D : D \text{ is a cluster in } \tilde{\mathcal{S}}\}$. The edge set Γ is defined as follows: For any cluster $C \in \mathcal{S}$, add a directed edge (C, D) , where D maximizes $|D' \cap C|$ over clusters D' of $\tilde{\mathcal{S}}$ breaking ties arbitrarily. Symmetrically, for cluster D of $\tilde{\mathcal{S}}$ add a directed edge (D, C) where C maximizes $|C' \cap D|$ over clusters C' of \mathcal{S} , breaking ties arbitrarily. Note that the out degree of all vertices of H is exactly 1. We will now define a bipartite matching on U using the following rule: If for some C, D , both $(C, D) \in \Gamma$ and $(D, C) \in \Gamma$, then match C to D , and call the pair (C, D) a *good match*. The remaining (unmatched) vertices are matched arbitrarily, and the corresponding pairs are called *bad matches*.

By the above claim, if (C, D) is a good match, then $\max\{|C \setminus D|, |D \setminus C|\} = O(\gamma^{1/3}|S|)$.

We now show that if (C, D) is a bad match then both $|C| = O(\delta^{1/3}|S|)$ and $|D| = O(\delta^{1/3}|S|)$. By symmetry, it suffices to show that $|C| = O(\delta^{1/3}|S|)$. Let C be a cluster of \mathcal{S} that is a member of a bad match. Let D be the unique cluster of $\tilde{\mathcal{S}}$ such that $(C, D) \in \Gamma$ and let C' be the unique cluster of \mathcal{S} such that $(D, C') \in \Gamma$. By the definition of a bad match we know that $C \neq C'$. By the above claim we have that both $|C \setminus D| = O(\delta^{1/3}|S|)$ and $|D \setminus C'| = O(\delta^{1/3}|S|)$, which implies $|C \setminus C'| = O(\delta^{1/3}|S|)$. But $C \cap C' = \emptyset$, therefore $|C| = O(\delta^{1/3}|S|)$. This concludes the proof of the lemma.

D Algorithm for High-Cost MFAST

Algorithm 3 query efficient PTAS for MFAST

Input: A graph $T=(V,A)$ an approximation parameter ε and assumed minimal cost parameter γ

Output: a permutation $\sigma : V \rightarrow [n]$ where $n = |V|$

For each $u \in V$ randomly select a sample $S^u = \{v_1^u, \dots, v_p^u\}$, where $p = O((\varepsilon\gamma)^{-2} \log(n))$ and the v_i^u 's are chosen from V with repetitions. Denote the ensemble $\{S^u : u \in V\}$ by \mathcal{S} . (This is the *verification* sample.)

Set S as a set of random i.i.d. vertices $S = \{v_1, \dots, v_s\}$ chosen with repetitions, where $s = O((\varepsilon\gamma)^{-2} \log(n))$. (This is the *enumeration* sample.)

Set $m = O((\gamma\varepsilon)^{-1})$ as the number of buckets.

For each possible m -bucket order of the vertices of S perform the following:

- Construct an LP as described in Section 5.2, producing a fractional m -bucket order that agrees with the bucketing of S .
- Solve the LP and round it as described in Section 5.1.

Pick the rounded solution whose approximated cost w.r.t. \mathcal{S} ($\text{cost}^{\mathcal{S}}(\cdot)$) is minimal, and output an arbitrary permutation extending it.
